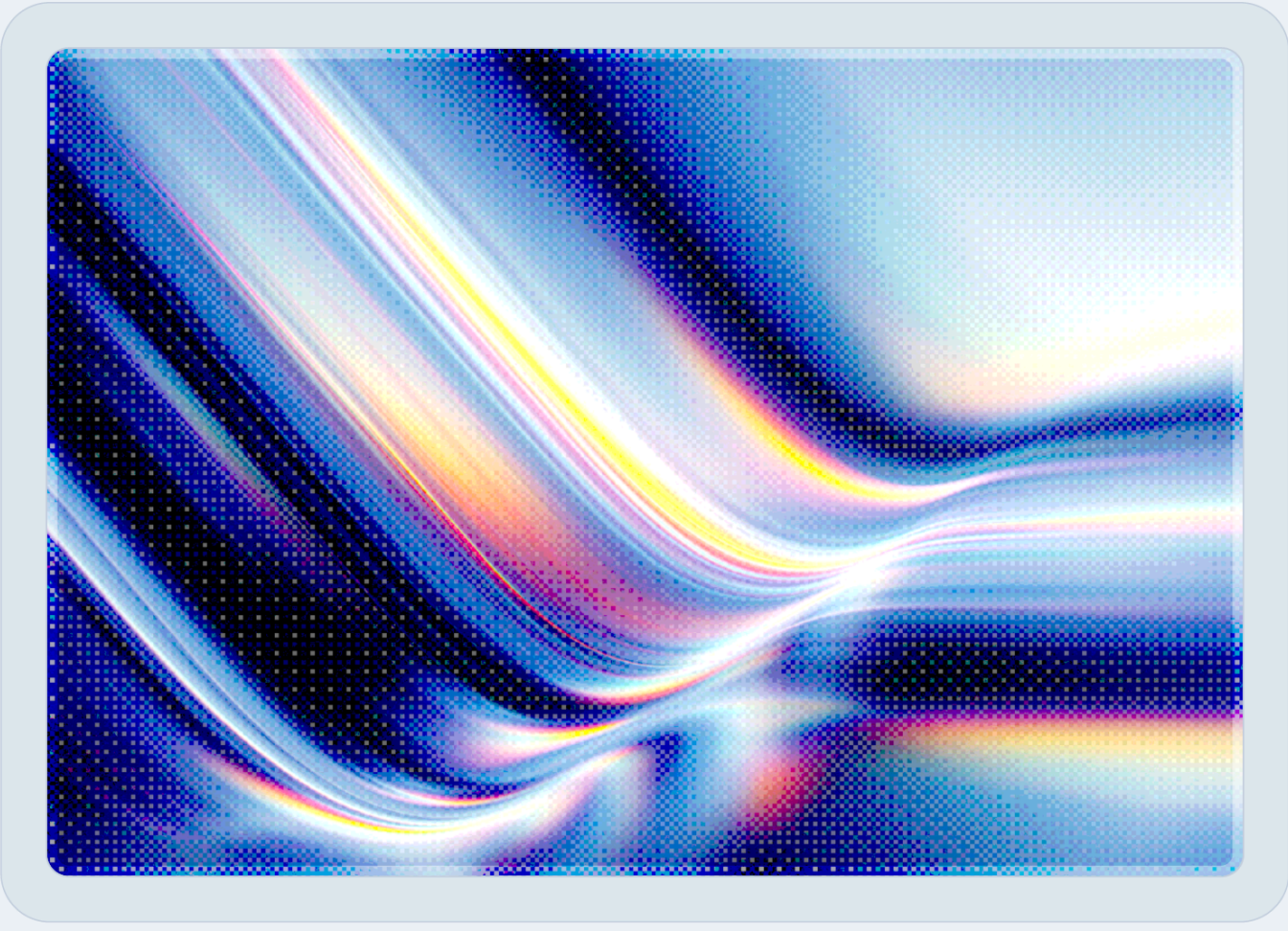
CSAI Foundation | Cloud Security Alliance

Inside the Enclave, Still Exposed: The NEAR AI SGLang Breach

Security Implications and Guidance for Confidential Inference

2026-11-11

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

On November 9, 2026, NEAR AI disclosed that an unauthorized party exploited a remote code execution (RCE) vulnerability in SGLang, the open-source inference engine running on its servers. The attacker gained code execution on one inference server and could have observed user requests processed on that server on November 8, including requests submitted in confidential mode [1]. The attacker then moved from that server to internal systems behind NEAR AI's deprecated Private Chat feature and downloaded names and email addresses for about 100 users. The attacker also retrieved chat history, settings, and logs for 18 agents on a legacy deployment that ran outside the confidential computing environment, and obtained API keys for two Cloud customers that had been written to logs [1][2][3].

The central lesson is one NEAR AI stated plainly in its own report: "Confidential computing protects data from the operator and the host. It does not protect against a vulnerability in the software running inside the environment" [1]. Hardware isolation and remote attestation answer two questions: whether the cloud operator can see the data, and whether the expected code is running. Neither answers whether that code is safe. NEAR AI's own account indicates that the vulnerable engine ran inside the confidential environment, although the company has not published the boundary layout. Where that is the case, the boundary keeps the host out but does nothing to keep out an attacker who arrives through the application itself, and it may also limit the operator's ability to observe the compromise.

Several further points follow for organizations that rely on confidential inference. An attestation report shows that a specific build is running; it does not show that the build is free of exploitable flaws, so exposure to a known-vulnerable inference engine is still exposure. SGLang has had a steady run of critical, often unauthenticated, RCE disclosures in 2026, and all three CERT/CC advisories discussed in this note reported no vendor response and no available patch at publication [5][6][9]. At the same time, the most consequential data loss in this incident came from systems outside or adjacent to the enclave: a deprecated feature, a legacy agent deployment, and secrets written to logs [1]. Buyers of confidential inference should therefore ask for evidence about the security of the software inside the enclave, such as its patch cadence, exposed attack surface, and egress controls, alongside the hardware attestation evidence.

Background

Confidential Inference and the NEAR AI Model

Confidential inference applies confidential computing to large language model (LLM) serving. In NEAR AI's design, models run on hardware that combines Intel Trust Domain Extensions (TDX) confidential virtual machines with NVIDIA GPUs operating in confidential computing mode. Prompts are carried over TLS and, according to NEAR AI, decrypted only inside the enclave, with memory "sealed from the host and operators" [7]. Each response can carry an attestation quote that binds it to the measured model and code. NEAR AI also publishes its runtime, verifier, and enclave images so that third parties can check what is running [7]. The value proposition is that neither the infrastructure provider nor NEAR AI's own operators can read user prompts.

This model has attracted downstream adopters. On October 15, 2026, Brave announced end-to-end encrypted chats in its Leo assistant, using the open-weight GLM 5.3 Flash model hosted by NEAR AI inside Intel TDX and NVIDIA TEEs [8]. Brave's design encrypts each prompt in the browser using Oblivious HTTP and HPKE. Brave's verification service checks both the Intel TDX and the NVIDIA attestations and confirms that NEAR-authored code matches public source, and the live attestation is re-checked against those approvals before traffic is relayed. Brave's gateway, for its part, strips user IP addresses so the TEE does not learn who is asking [8]. Brave's write-up was candid about residual trust. It noted that the third-party model runtime is not rebuilt from source but is instead pinned through a hardware-attested deployment log, and that the chain of trust depends on Intel, NVIDIA, dstack, Sigstore, GitHub, Hugging Face, and public DNS [8]. NEAR AI's incident report does not identify any Brave users as affected, and Decrypt reported the same [2].

SGLang's 2026 Vulnerability Record

SGLang is a popular open-source framework for serving LLMs and multimodal models. Over 2026 it has accumulated a notable series of vulnerability disclosures, many of them unauthenticated RCE flaws caused by unsafe deserialization. In April, CERT/CC published VU#915947 for CVE-2026-5760, a CVSS 9.8 flaw in which a malicious GGUF model's chat template is rendered in an unsandboxed Jinja2 environment when a request reaches the `/v1/rerank` endpoint [4][9][10]. CSA's AI Safety Initiative also covered CVE-2026-3059 and CVE-2026-3060, two CVSS 9.8 pickle deserialization flaws in SGLang's ZeroMQ broker and disaggregation modules [11]. Those two are reachable mainly in multi-node or distributed configurations [11]. The pattern itself is older. In late 2025, Oligo Security's "ShadowMQ" research found that unsafe ZeroMQ pickle handling had spread across several inference frameworks, and it reported SGLang as having only an incomplete fix at the time [12].

The disclosures continued through mid-2026. CERT/CC's VU#777338, published May 18, covered two RCE flaws and one path traversal flaw, two of them in SGLang's multimodal generation runtime and one, CVE-2026-7304, a `dill.loads()` RCE reachable when the optional custom logit processor is enabled. Another, CVE-2026-7301, arises because a scheduler socket binds to all interfaces by default and passes incoming messages to `pickle.loads()` [5]. On July 30, CERT/CC's VU#281278 described six more vulnerabilities [6]. These were unauthenticated RCE through `/load_lora_adapter_from_tensors`, RCE via a `torch.load` fallback in `/update_weights_from_disk`, a sandbox escape in the optional dumper subsystem, server-side request forgery and local file read via `image_url` input, credential leakage via `/server_info`, and model weight exfiltration when no API keys are configured [6]. As with VU#915947, both later CERT/CC notes reported that the vendor had not provided a statement and that no patch was available at publication [5][6][9]. NEAR AI has not said which SGLang vulnerability was exploited, whether it matches any of these published CVEs, or which version was affected [1][3].

Incident Timeline and Scope

The table below summarizes what NEAR AI and subsequent reporting have established. Items marked as not disclosed reflect gaps in the public record as of this writing.

Element	Publicly reported detail	Source
Initial access	RCE vulnerability in SGLang on one inference server	[1]
Confidential requests	May have been observable on the accessed server on November 8, 2026; volume not quantified	[1][13]
Lateral movement	From the inference server to internal systems behind deprecated Private Chat	[1]
User data	About 100 users: names, emails, and for some, plan and balance information	[1][2]
Agent data	18 legacy agents outside the TEE: chat history, settings, logs	[1][2]
Credentials	Two Cloud customers' API keys exposed in logs; both revoked	[1][2]
Use of exposed API keys	Not disclosed	[2]

Element	Publicly reported detail	Source
Stored conversations	No indication that stored Private Chat conversations or uploaded files were accessed	[1]
Exploited CVE or version	Not disclosed	[1][3]
Detection date, dwell time	Not disclosed	[1]
Disclosure	Incident report published November 9, 2026	[1]

NEAR AI's containment steps included revoking all active sessions and administrative credentials, replacing all agent tokens, rotating remaining credentials, taking Private Chat offline, deploying an SGLang fix across all inference servers, completing a forensic review, notifying affected users, and reporting the vulnerability to the SGLang project [1]. NEAR AI also reported that NEAR AI Cloud access has been restored [1].

Security Analysis

What the Enclave Did and Did Not Protect

To read this incident correctly, separate the threat that confidential computing is designed to address from the threat that materialized. TEEs such as Intel TDX protect a workload's memory from a privileged host: the hypervisor, the cloud operator, and administrators with physical or root access to the machine. Remote attestation then lets a relying party verify cryptographically that a particular measured software stack is running inside that protected environment. Both properties are about who sits outside the boundary. Neither constrains what the software inside the boundary does when it receives a malicious input.

An inference engine like SGLang must, by design, see plaintext prompts in order to tokenize and process them. NEAR AI's report states that the vulnerability "let the party run code inside one inference server and access requests in processing on that server, including confidential ones" [1][3]. This suggests that the attacker's code ran in the same context where plaintext existed, which is inside the confidential boundary. NEAR AI's public report does not fully specify the boundary layout. From the attacker's perspective, however, the enclave appears to have functioned as intended: it kept the host out, and it had no reason to

keep out an attacker who arrived through the application's own network interface. A request that arrives over an authenticated, attested channel and is then exfiltrated by malicious code within the same confidential VM is, from the hardware's point of view, legitimate processing.

Attestation Proves Identity, Not Safety

This incident also clarifies a common misreading of attestation. An attestation quote is evidence that the measured components match expected values. If a vulnerable SGLang build is part of the measured image, attestation will faithfully report that the vulnerable build is running, and a verifier will accept it because it matches the published reference. Brave's architecture shows the nuance well: the third-party model runtime is pinned through a hardware-attested deployment log rather than rebuilt from reviewed source [8]. That control guarantees consistency and transparency, but it cannot by itself make an exploitable runtime safe.

Attestation can still support vulnerability management if relying parties use it that way. A verifier could reject measurements that correspond to known-vulnerable component versions, much as certificate validation rejects revoked certificates. Publicly documented verification flows, including the Brave design described above, check whether a measurement matches the vendor's published reference [8]; we are not aware of widely deployed verifiers that also evaluate whether the referenced build carries outstanding critical CVEs. Treating attestation measurements as inputs to a vulnerability policy, rather than as a binary trust signal, appears to be an underdeveloped practice. This incident suggests it deserves attention.

The Inference Engine as High-Value Attack Surface

SGLang's 2026 record shows a recurring weakness class rather than a one-time defect. Unsafe deserialization through `pickle`, `dill`, or `torch.load` fallbacks appears in the ZeroMQ broker, the disaggregation module, the multimodal scheduler, the custom logit processor, the LoRA adapter loading path, and the weight update path [5][6][11]. Several of these are administrative or operational endpoints, such as weight updates, adapter loading, and server information, that exist for operator convenience but share a process, and in default configurations a network interface, with user-facing inference. CVE-2026-7301's default bind to `0.0.0.0` illustrates how a deployment can expose internal control channels without any deliberate misconfiguration [5].

For confidential inference operators the implication is direct. The inference engine necessarily sits inside the trust boundary, since it must see plaintext; it exposes a broad attack surface; and, based on this year's CERT/CC notes, a component whose upstream maintainers have not always been reachable during coordinated disclosure [5][6]. An operator who runs this software inside an enclave inherits responsibility

for its hardening. That includes disabling unused endpoints, setting `SGLANG_USE_PICKLE_IPC` to false as CERT/CC recommended, and keeping control interfaces off any network path reachable by tenants [6].

Blast Radius Beyond the Enclave

A second lesson concerns what happened after initial access. The confirmed exfiltration of user records, agent chat histories, and API keys came from systems NEAR AI described as deprecated, legacy, or outside the TEE [1]. The inference server became a pivot point into adjacent infrastructure. This pattern is familiar from conventional breaches, and it suggests that the inference host had network reachability to internal systems that it did not strictly need. Customer API keys appearing in logs points to a separate, common hygiene gap.

These findings matter because product messaging for confidential inference, including NEAR AI's own, centers on the enclave [7], while an organization's overall exposure depends on every system that touches user identity, credentials, and history. A strong enclave next to weak legacy systems produces a mixed security posture. A user who sees TEE protection emphasized in product messaging may reasonably, but incorrectly, assume that the guarantee extends to account data and agent state.

Disclosure Quality

NEAR AI's report was published on November 9, one day after the date on which it says confidential requests may have been observable; the start of the intrusion and the detection time were not disclosed. It acknowledged the architectural limit of confidential computing directly, and it maintained the public availability of its stack's code and configuration [1]. Those choices are constructive. The report leaves material questions open, however: the specific SGLang vulnerability or CVE, the affected version, detection time and dwell time, whether the compromised server was itself a TEE-protected machine, how many confidential requests may have been observed, and whether attestation measurements changed after the fix [1][3][13]. For a service whose value depends on verifiability, publishing the before-and-after attestation measurements and the patched component versions would let relying parties confirm remediation independently.

Recommendations

Immediate Actions

Organizations that use NEAR AI Cloud, or products built on it, should confirm whether they received a direct notification from NEAR AI and rotate any NEAR AI Cloud API keys as a precaution, even if they were not among the two customers named [1]. Because NEAR AI warned users to treat unsolicited messages that reference their account or balance with caution, security teams should brief affected users on phishing risk [1][2].

Any organization that runs SGLang, whether inside a TEE or not, should inventory its deployments and versions and review exposure against the CERT/CC advisories VU#915947, VU#777338, and VU#281278 [5][6][9]. Where patches are unavailable or unconfirmed, apply CERT/CC's mitigations. Restrict network access to the service and keep scheduler, broker, and administrative interfaces off untrusted networks [5] [6]. In addition, disable unused endpoints such as weight-update and adapter-loading routes, and set `SGLANG_USE_PICKLE_IPC` to false [6]. Teams should also make sure SGLang is never started without API keys and that `/server_info` is not reachable by untrusted callers, given the credential leakage and weight exfiltration paths described in CVE-2026-15977 and CVE-2026-15978 [6].

Short-Term Mitigations

Over the next several weeks, operators of confidential inference should apply defense in depth inside the trust boundary rather than treating the enclave as the perimeter. The table below maps the controls most directly relevant to this incident to the weakness each one addresses.

Control	Implementation	Incident weakness addressed
Control-plane separation	Separate the user-facing inference API from administrative and control-plane endpoints at the process and network level, and authenticate every internal channel	Administrative endpoints sharing a process and interface with inference
Egress filtering	Apply strict egress filtering from inference hosts so a compromised engine cannot reach internal databases, legacy services, or arbitrary internet destinations	Lateral movement from the inference server to internal systems

Control	Implementation	Incident weakness addressed
Least privilege in the confidential VM	Run the engine with seccomp profiles, read-only filesystems, and no shell or package tooling in production images	Attacker code execution inside the inference server
Log redaction	Scrub logs of API keys, session tokens, and prompt content, and route logs out of the enclave only after redaction	Customer API keys written to logs
Full decommissioning	Retire deprecated features and legacy deployments completely, including their credentials, data stores, and network paths, rather than leaving them reachable	Data loss from deprecated Private Chat and legacy agents

Taken together, these controls treat a compromise of the inference engine as an expected failure mode and limit what an attacker can reach from it. Relying parties that verify attestation, such as browser vendors, enterprise gateways, and API consumers, should extend their verification policies to deny-list measurements tied to known-vulnerable component versions. They should also ask providers to publish component version manifests alongside attestation references.

Strategic Considerations

Over the longer term, the industry would benefit from a clearer shared vocabulary for what confidential inference does and does not guarantee. Product claims should distinguish protection from the operator and host, which TEEs provide, from protection against compromise of the software inside the TEE, which they do not. Procurement and assurance processes for AI services should evaluate the inference software supply chain with the same rigor applied to the hardware root of trust. That evaluation should cover upstream maintainer responsiveness, patch latency, deserialization practices, and default network exposure.

Inference framework maintainers and their downstream commercial users should also consider structural changes to the weakness class behind SGLang's 2026 vulnerabilities. Possible changes include eliminating `pickle` and `dill` from network-facing interprocess communication, defaulting to safe tensor formats, and binding control sockets to loopback by default. Where upstream coordination is slow, as CERT/CC reported for several SGLang advisories, organizations that depend commercially on these frameworks may need to fund or contribute security engineering directly [5][6]. VU#281278 does note

that SGLang maintainers have begun migrating interprocess communication to msgpack, a step in this direction [6]. Finally, confidential inference providers may want to explore architectures that shrink what must be trusted inside the boundary. Examples include isolating tokenization and request handling from the model runtime, or using multiple mutually isolated enclaves, so that a single engine compromise does not expose all requests in flight.

CSA Resource Alignment

CSA's AI Safety Initiative has tracked SGLang's vulnerability trajectory throughout 2026. The research note "AI Infrastructure Under Attack: SGLang RCE and MCP Memory Poisoning" analyzed CVE-2026-3059 and CVE-2026-3060 and identified unsafe deserialization in distributed inference components as a systemic risk [11]. The companion note "SGLang CVE-2026-5760: RCE via Poisoned GGUF Model Files" examined the model-file vector [10]. The NEAR AI incident is consistent with the risk those notes described. Although the specific vulnerability has not been disclosed, the incident confirms that SGLang RCE flaws are being exploited against production services, including one where users expected heightened privacy.

CSA's white paper "Healthcare Confidential Computing and the Trusted Execution Environment" explains TEEs, hardware-based and remote attestation, and how these technologies compare with alternatives such as HSMs, TPMs, and homomorphic encryption for protecting data in use [14]. Its framing of the TEE as protection for data during processing is the right baseline for this incident. This note adds a qualification: the protection holds against outside observers, not against compromised code running inside the TEE. Organizations applying that paper's guidance to AI workloads should pair TEE adoption with application-layer hardening of the inference software placed inside the boundary.

CSA's "Data Security within AI Environments" maps data security controls across the AI lifecycle to the AI Controls Matrix and identifies gaps in coverage [15]. The NEAR AI incident falls squarely within its scope. User prompts in processing, agent chat histories, and credentials written to logs are all AI-environment data that require protection across the lifecycle, not only at the inference step. Finally, the AI Controls Matrix (AICM) v1.1 provides the control vocabulary for the recommendations above [16]. Its Threat and Vulnerability Management domain addresses patching and vulnerability tracking for inference engines. Its Application and Interface Security domain addresses endpoint exposure and deserialization safety. Its Identity and Access Management domain addresses credential rotation and least privilege for inference hosts. Its Data Security and Privacy domain addresses log redaction and retention.

References

[1] NEAR AI. "NEAR AI Security Incident Report: SGLang Vulnerability." NEAR AI, November 9, 2026.

[2] Lanz, Jose Antonio. "Near AI Breach Exposes User Details and Data From 18 AI Agents." Decrypt, November 10, 2026.

[3] Park, Danny. "NEAR AI says SGLang vulnerability exposed user account details in security breach." The Block, November 10, 2026.

[4] The Hacker News. "SGLang CVE-2026-5760 (CVSS 9.8) Enables RCE via Malicious GGUF Model Files." The Hacker News, April 20, 2026.

[5] CERT/CC. "VU#777338: SGLang contains two remote code execution and one path traversal vulnerability." CERT Coordination Center, May 18, 2026.

[6] CERT/CC. "VU#281278: SGLang contains six different vulnerabilities including RCE, data exfiltration, and credential disclosure." CERT Coordination Center, July 30, 2026.

[7] NEAR AI. "Private AI inference you can verify." NEAR AI, accessed November 11, 2026.

[8] Brave. "Another Brave User-First AI Win: Leo adds End-to-end Encryption, Oblivious HTTP, and Verifiable Confidential Computing." Brave, October 15, 2026.

[9] CERT/CC. "VU#915947: SGLang is vulnerable to remote code execution when rendering chat templates from a model file." CERT Coordination Center, April 20, 2026.

[10] Cloud Security Alliance AI Safety Initiative. "SGLang CVE-2026-5760: RCE via Poisoned GGUF Model Files." CSA, April 22, 2026.

[11] Cloud Security Alliance AI Safety Initiative. "AI Infrastructure Under Attack: SGLang RCE and MCP Memory Poisoning." CSA, April 24, 2026.

[12] The Hacker News. "Researchers Find Serious AI Bugs Exposing Meta, Nvidia, and Microsoft Inference Frameworks." The Hacker News, November 14, 2025.

[13] Morrison, Maisie. "Near AI Says Hacker May Have Seen Confidential AI Requests in Server Breach." Blockonomi, November 11, 2026.

© 2026 Cloud Security Alliance. AI-assisted Rapid Research.

[14] Cloud Security Alliance. "Healthcare Confidential Computing and the Trusted Execution Environment." CSA, 2025.

[15] Cloud Security Alliance. "Data Security within AI Environments." CSA, 2025.

[16] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." CSA, 2026.